REJ10J1725-0100

Everywhere you imagine. RENESAS

M16C R8C FoUSB/UART Debugger

User Manual

Renesas Microcomputer Development Environment System
R8C Family R8C/2x Series
Notes on Connecting R8C/2A, R8C/2B, R8C/2C, R8C/2D

Rev.1.00
Issued : July 15, 2008

Renesas Technology
www.renesas.com

Active X, Microsoft, MS-DOS, Visual Basic, Visual C++, Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.

IBM and AT are registered trademarks of International Business Machines Corporation.

Intel and Pentium are registered trademarks of Intel Corporation.

Adobe and Acrobat are registered trademarks of Adobe Systems Incorporated.

All other brand and product names are trademarks, registered trademarks or service marks of their respective holders.

Keep safety first in your circuit designs!

- Renesas Technology Corporation and Renesas Solutions Corporation put the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Renesas Technology product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation, Renesas Solutions Corporation or a third party.
- Renesas Technology Corporation and Renesas Solutions Corporation assume no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation and Renesas Solutions Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation, Renesas Solutions Corporation or an authorized Renesas Technology product distributor for the latest product information before purchasing a product listed herein. The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation and Renesas Solutions Corporation assume no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors. Please also pay attention to information published by Renesas Technology Corporation and Renesas Solutions Corporation by various means, including the Renesas home page (http://www.renesas.com).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation and Renesas Solutions Corporation assume no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Renesas Technology semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation, Renesas Solutions Corporation or an authorized Renesas Technology product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Renesas Technology Corporation and Renesas Solutions Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination. Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Renesas Technology Corporation or Renesas Solutions Corporation for further details on these materials or the products contained therein.

Table of Contents

1. Connecting the Target Board to the User System 4
2. Preparing the M16C R8C FoUSB/UART Debugger 6
3. Memory Map When Using the R8C UART Debugger 19
4. Monitor Program Occupied Area 21
5. Notes on Using the R8C UART Debugger 21
   - 5.1. Changing communication speed and restarting the R8C UART debugger after the R8C UART debugger is done 21
   - 5.2. User program ID code 21
   - 5.3. User program download area 22
   - 5.4. Frequency characteristics 25
   - 5.5. Limitations of SFR operations 26
   - 5.6. Limitations on stop mode or wait mode 26
   - 5.7. Watchdog timer 27
   - 5.8. Real-time operation of user program 27
   - 5.9. Executing anomalistic steps 28
   - 5.10. Limitations on peripheral functions 29
   - 5.11. Limitations on flag register 29
   - 5.12. Operation on peripheral I/O during a break 29

1. Connecting the Target Board to the User System

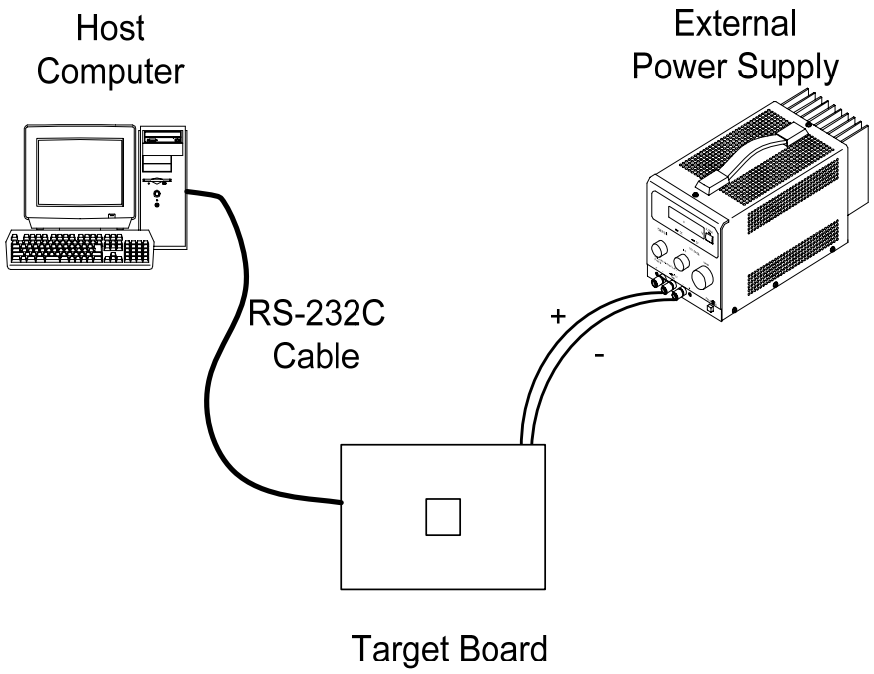


Figure 1 Connecting the Target Board to the User System

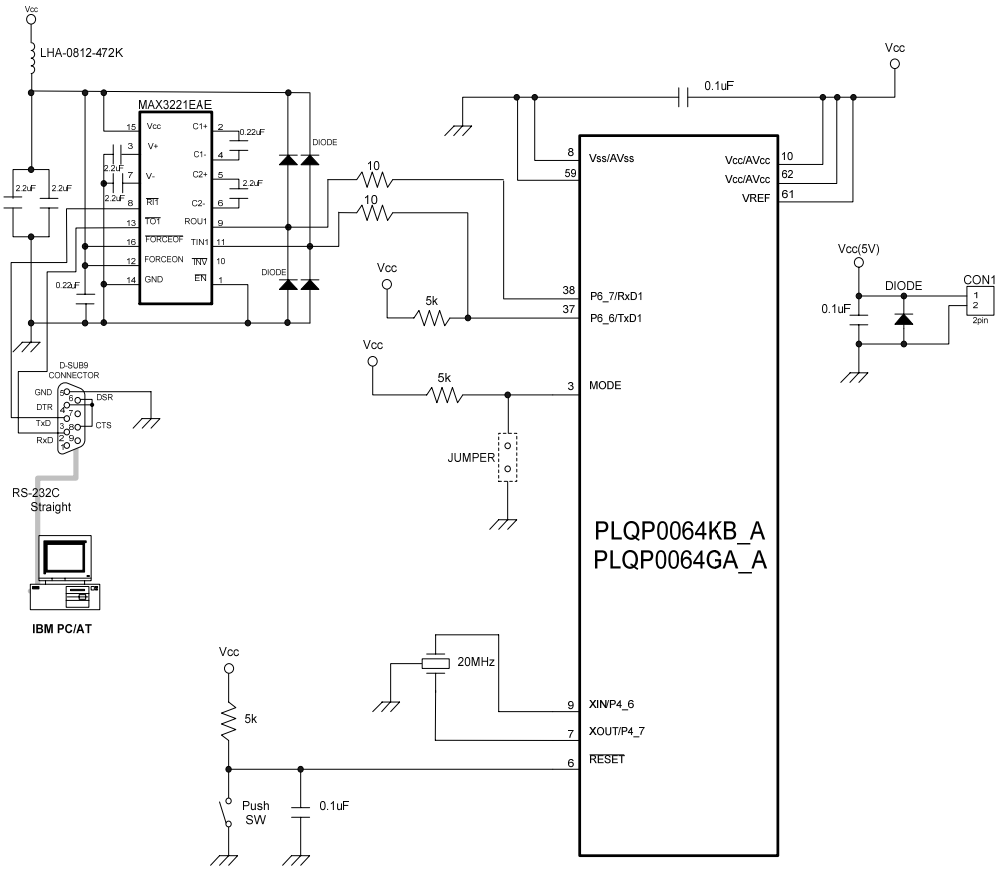


Figure 2 Circuit Using the RS-232C Cable (R8C/2A, 2B Groups)

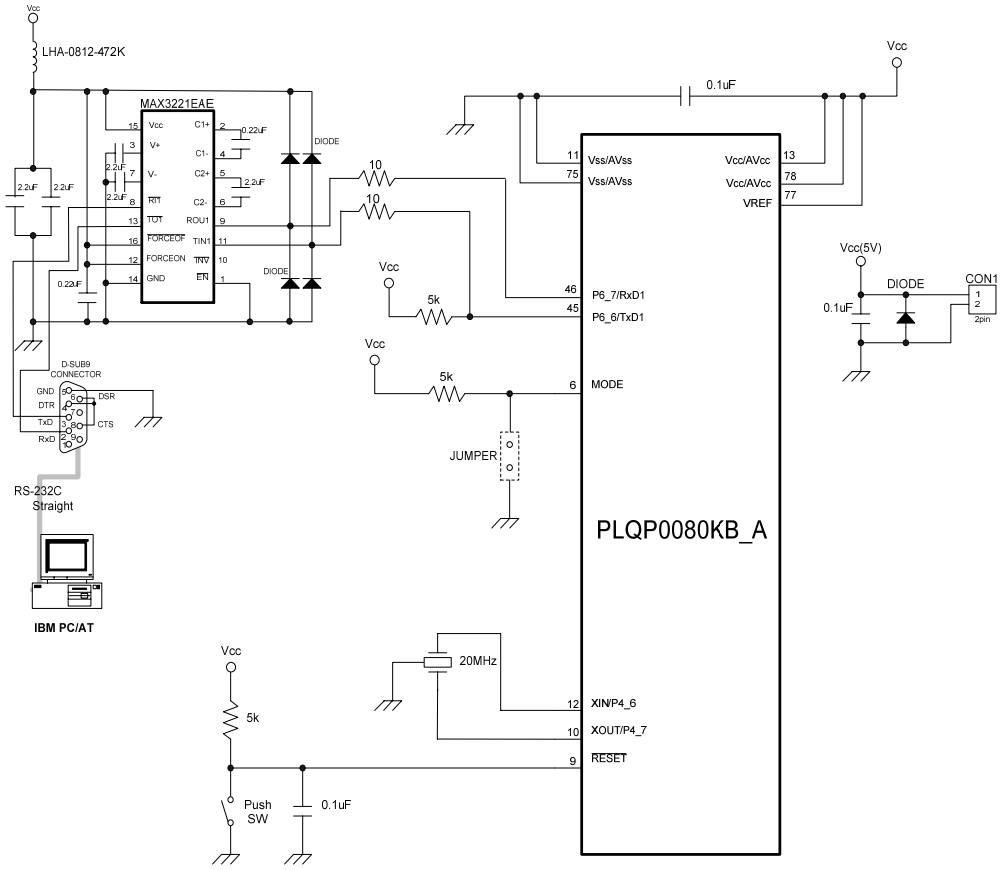


Figure 3 Circuit Using the RS-232C Cable (R8C/2C, 2D Groups)

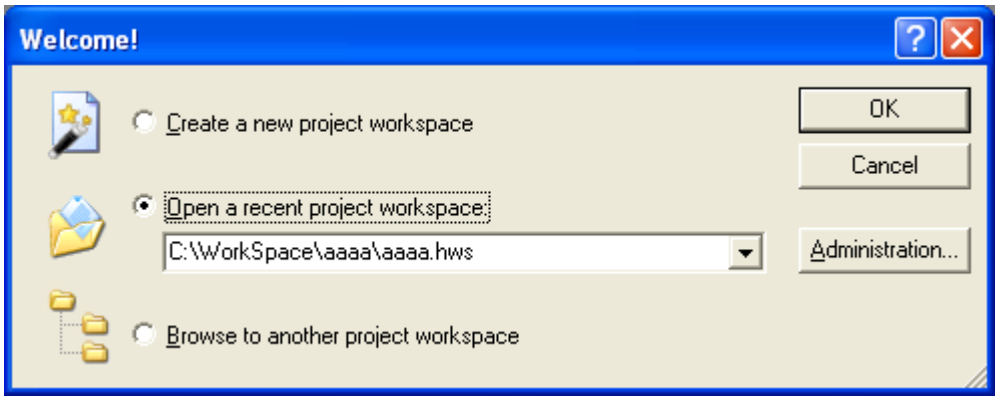
2. Preparing the M16C R8C FoUSB/UART Debugger

The M16C R8C FoUSB/UART debugger (hereinafter R8C UART debugger) is used for debugging by connecting an RS-232 cable between the host computer and the target board.

A monitor program comes bundled and is automatically installed with the R8C UART debugger software.

For the R8C Family, the monitor program is automatically programmed when starting the R8C UART debugger. The monitor program does not need to be preprogrammed with an M16C FlashStarter or other flash programmer.

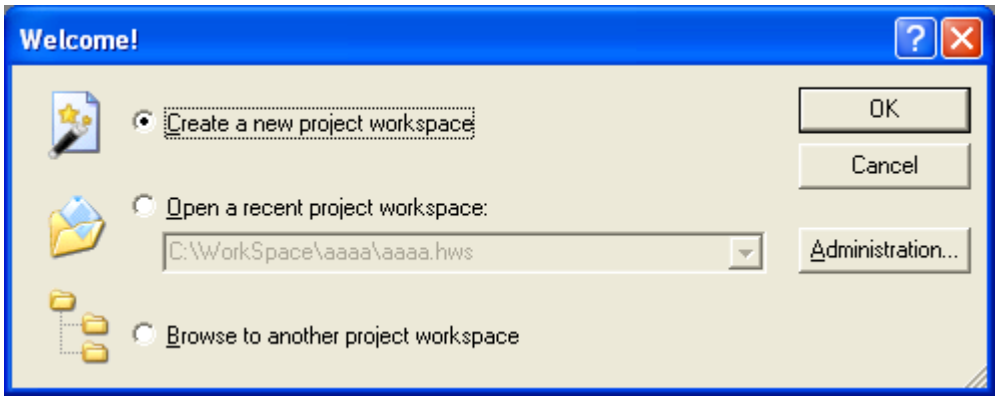
a) To start the High-performance Embedded Workshop (HEW), go to “Start”, “Programs”, “Renesas”, “High-performance Embedded Workshop”, and “High-performance Embedded Workshop”. You will see the following “Welcome!” dialog box.



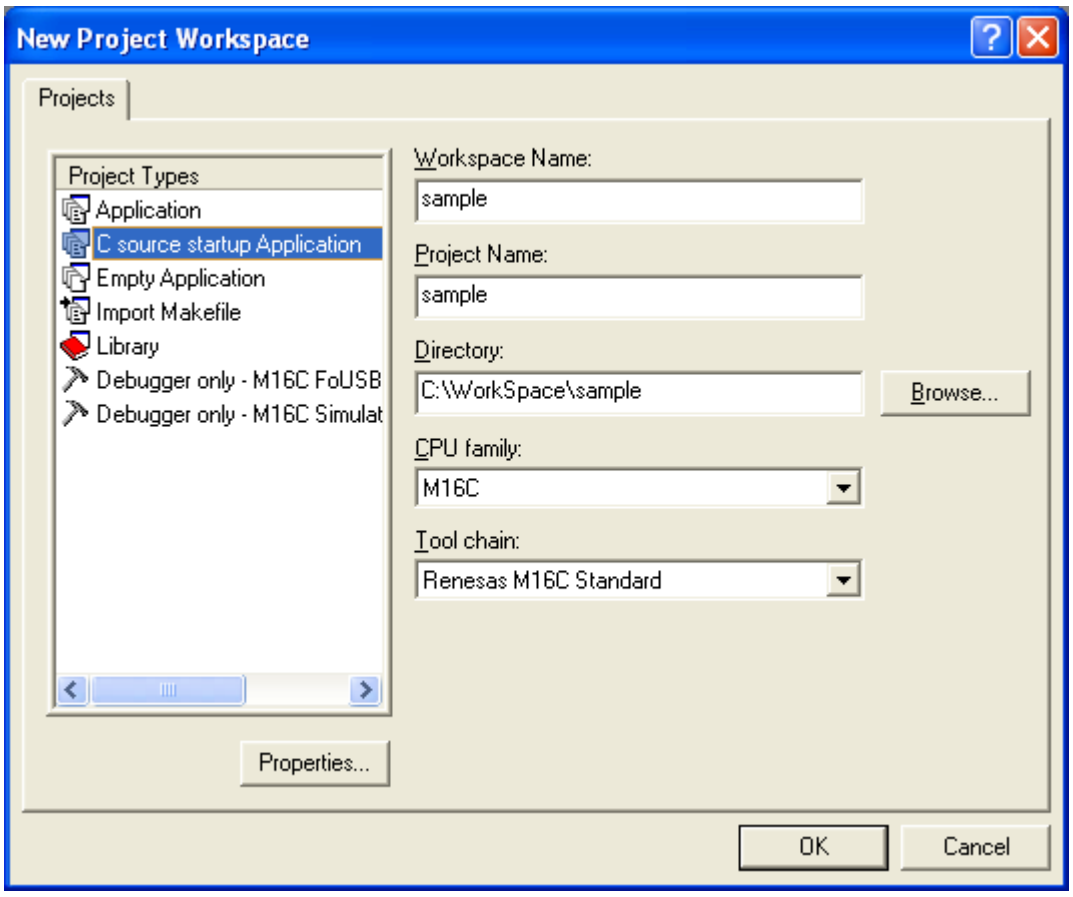
- [Create a new project work space]
  Select this when creating a new workspace.
- [Open a recent project workspace]
  Select this when using an existing workspace. Previous workspaces are displayed in the pull-down menu.
- [Browse to another project workspace]
  Select this when using an existing workspace, and there are no previous workspaces listed in the pull-down menu.

After selecting an existing workspace and pressing [OK], the screen under letter t) (Page 16) is displayed.

b) Select “Create a new project workspace”. Press [OK].



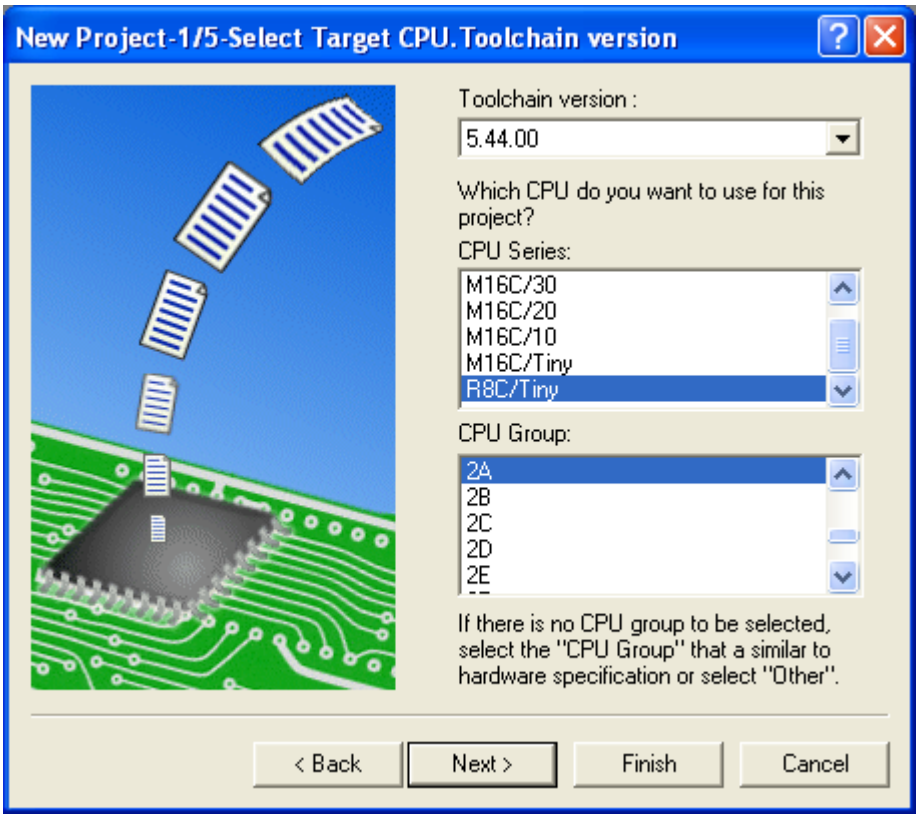
c) The Project Generator starts. When a toolchain is preinstalled, the following screen is displayed.



- [Workspace Name]
  Enter a name to create a new workspace. “sample” has been input as an example.
- [Project Name]
  Enter a project name. If the project name is to be the same as the workspace name, it is not necessary to input anything in the [Project Name] field.

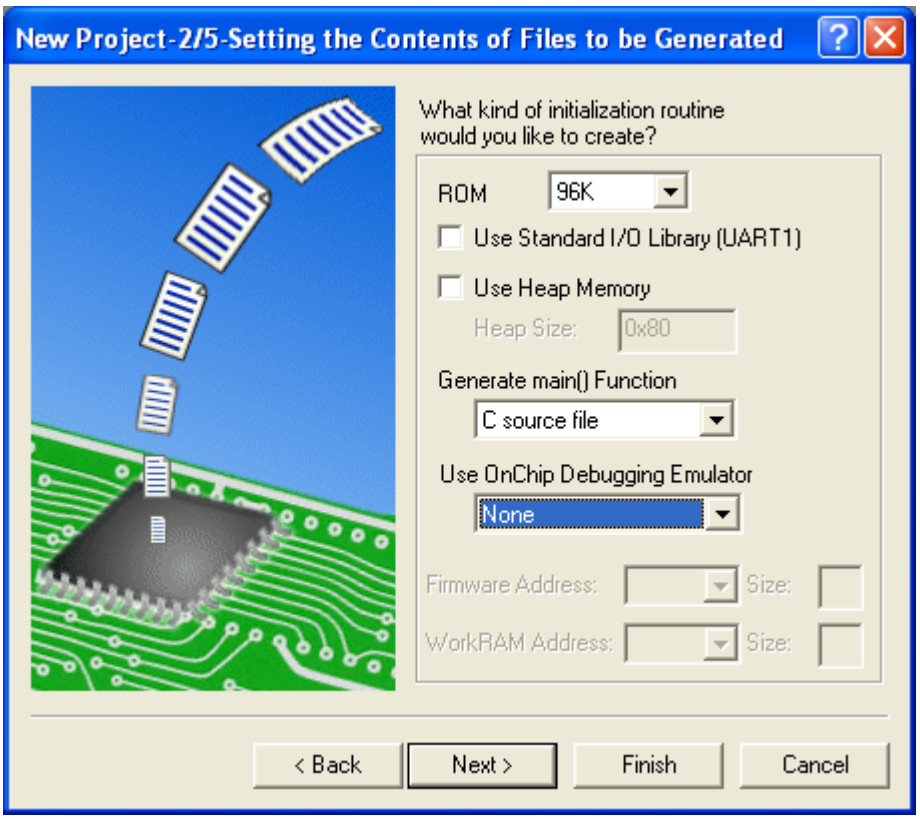
- [CPU family]
  Select the applicable CPU family. In this case, M16C has been selected for the R8C Family.
- [Toolchain]
  When using a toolchain, select the applicable toolchain name. When not using a toolchain, select “None”.
- [Project Types] Window
  Select a project type. The project type which supports the C start-up has been selected as an example.

d) Set the toolchain version.



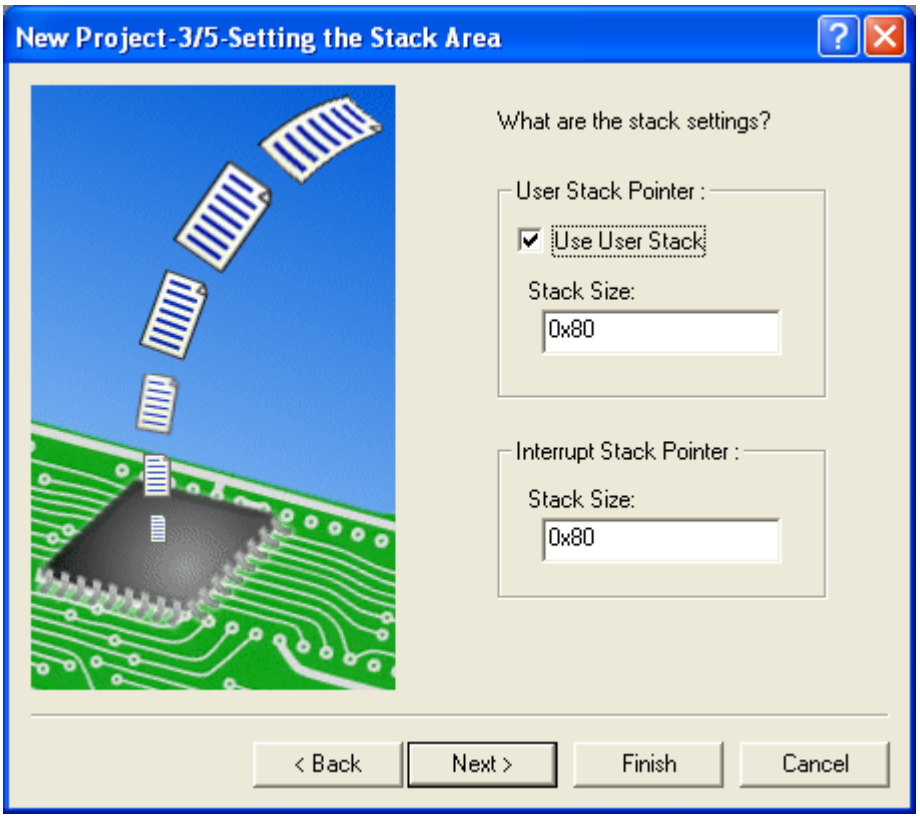
Select the “Toolchain Version”, “CPU Series”, “CPU Group” and then press [Next]. In this example, R8C/2A Group is selected.

e) Select the MCU ROM size and other related settings.



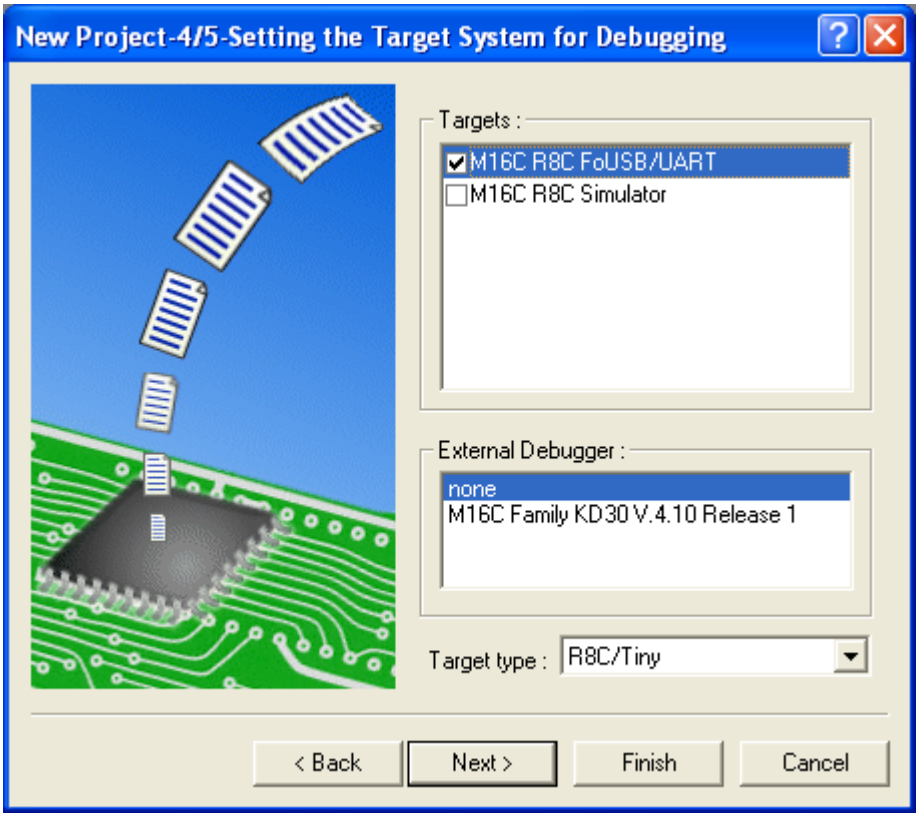
Select the MCU ROM size and heap size. Select “None” for the on-chip debugging emulator. And press [Next].

f) Set the stack.



Set the “Stack Size” and press [Next].

g) When the toolchain setting is complete, the following screen is displayed.

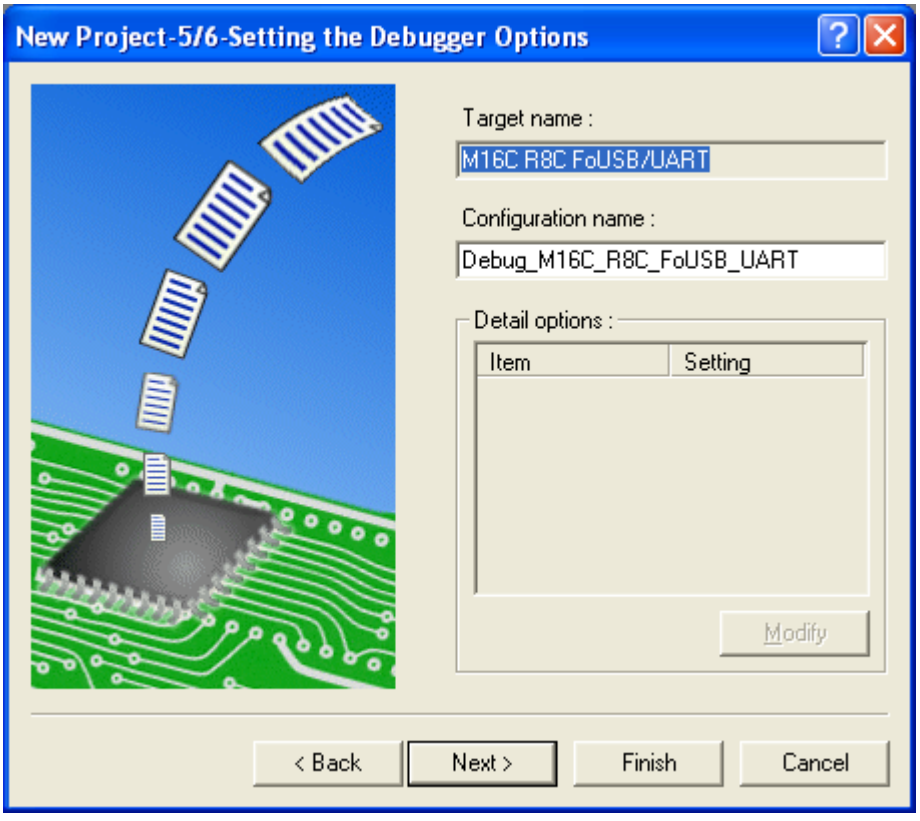


Select “M16C R8C FoUSB/UART” and press [Next].

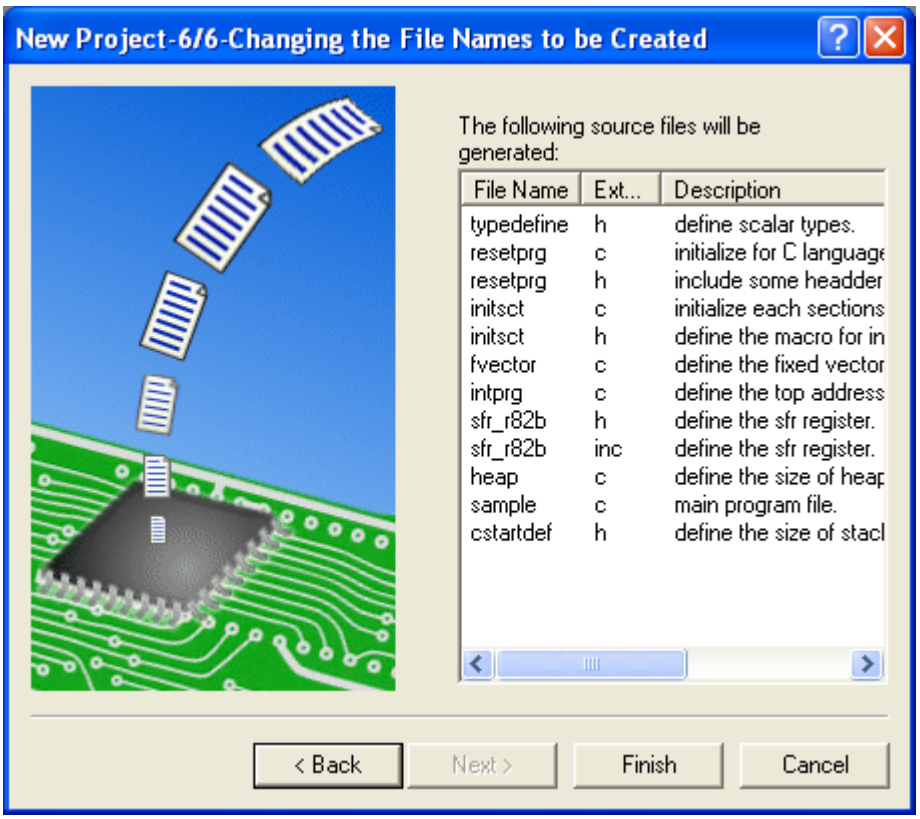
If necessary, select other targets.

h) Set the configuration file name.

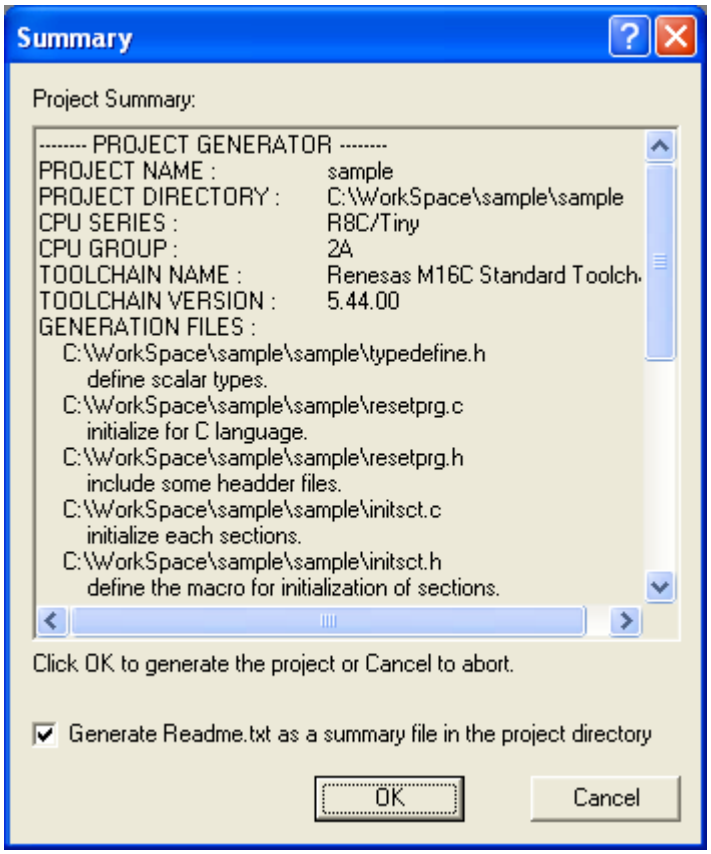
Configurations are the build option settings (e.g., output of debug information or optimization) having their own names. The term "configuration" can also be referred to as "build configuration".



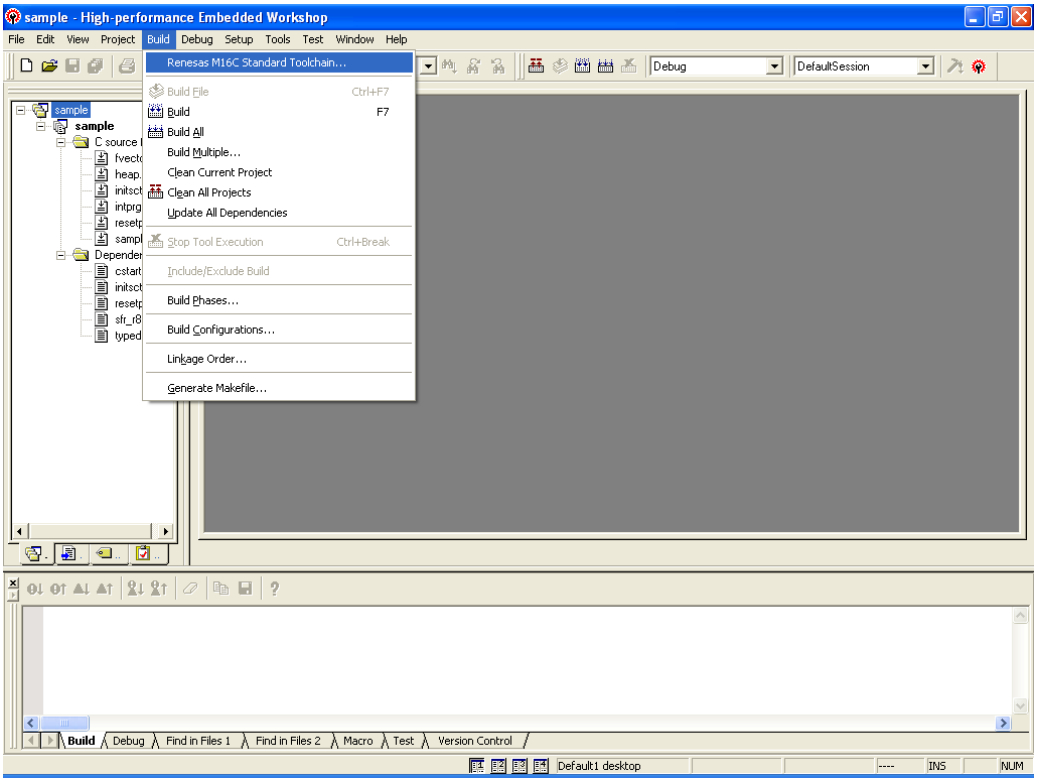
i) Verify the file names to be generated.


New Project-6/6-Changing the File Names to be Created
The following source files will be generated:
File Name Ext... Description
typedefine h define scalar types.
resetprg c initialize for C language
resetprg h include some headder
initsct c initialize each sections
initsct h define the macro for in
fvector c define the fixed vector
intprg c define the top address
sfr_r82b h define the sfr register.
sfr_r82b inc define the sfr register.
heap c define the size of heap
sample c main program file.
cstartdef h define the size of stacl
< Back
Next >
Finish
Cancel

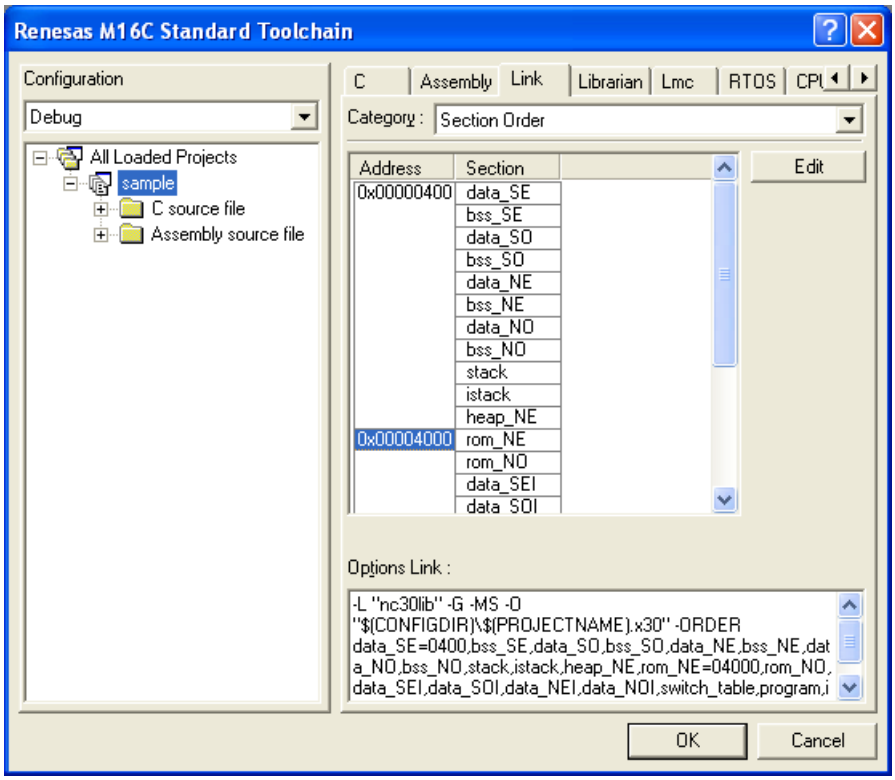
j) The figure below displays the files generated by the High-performance Embedded Workshop based on the above settings. Pressing [OK] launches High-performance Embedded Workshop.



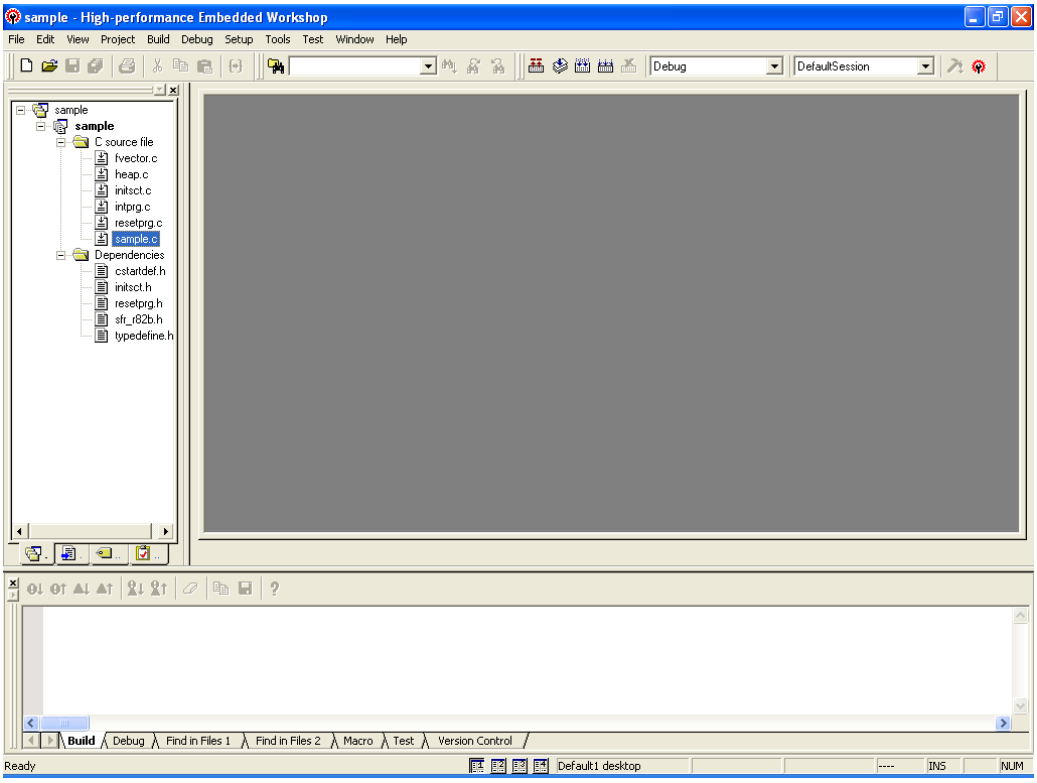
k) Check the section address. Under the “Build” menu, select “Renesas M16C Standard Toolchain”.



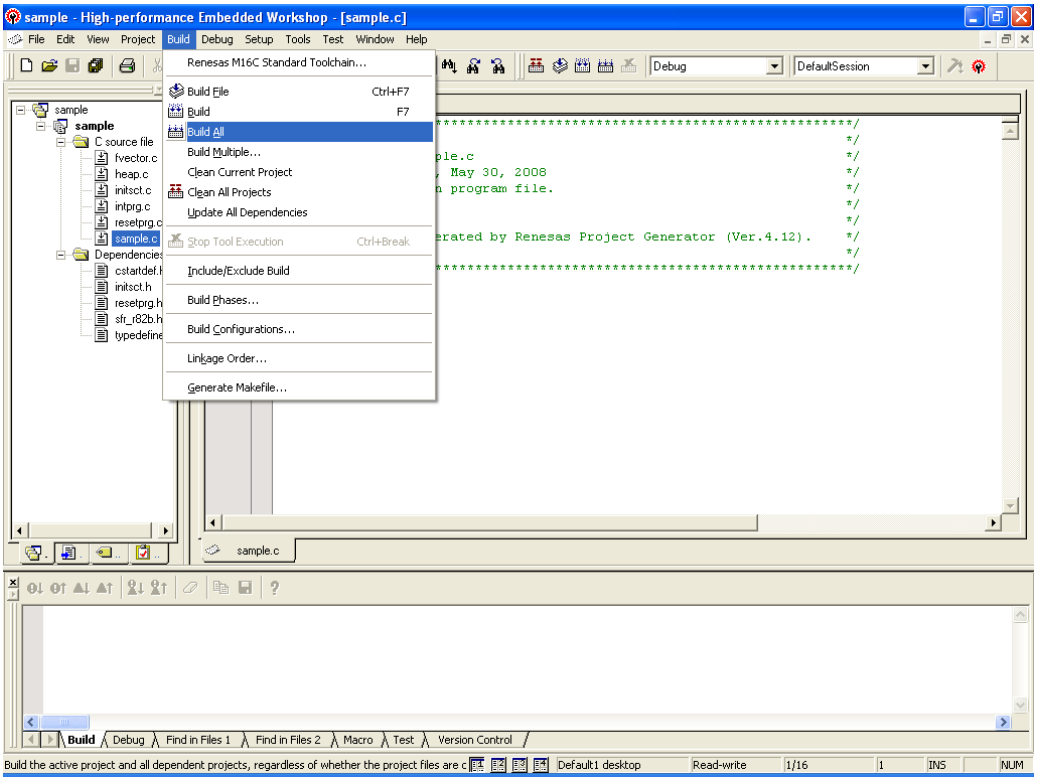
l) “Renesas M16C Standard Toolchain” is displayed. Click on the “Link” tab and select “Section Order” from the “Category” pull-down menu. Make sure the section start address does not overlap with the monitor program occupied area shown in Table 1. If there is any overlap, edit the address value.



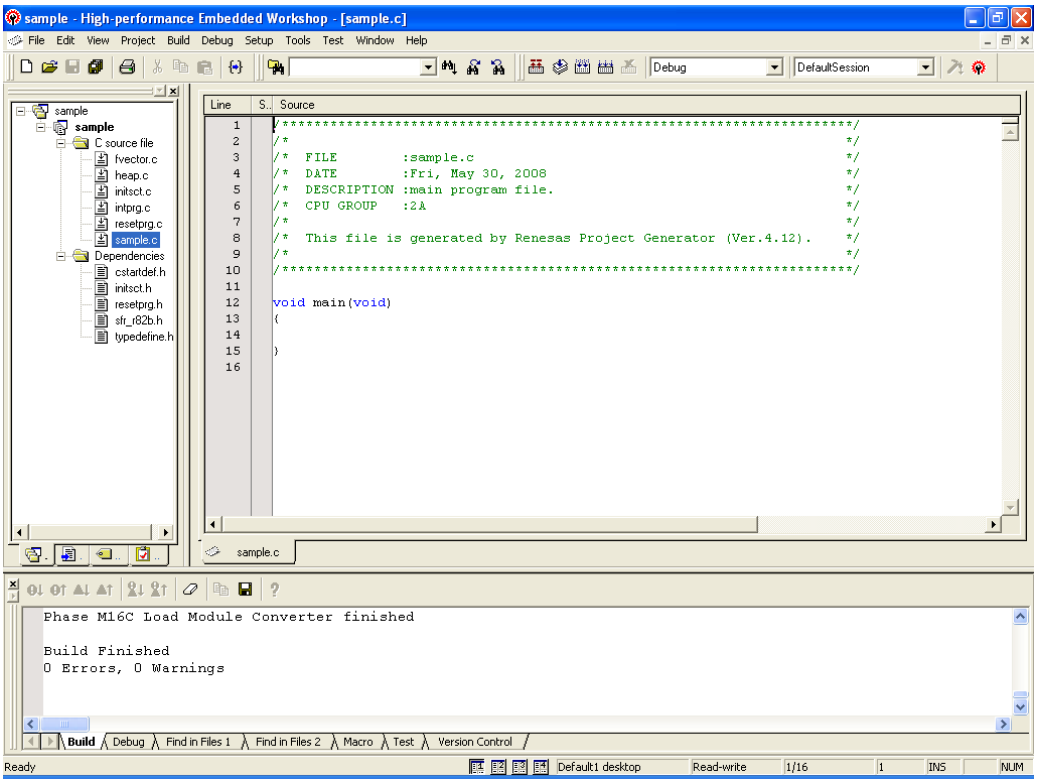
m) Double-click the source program to launch the program editor.



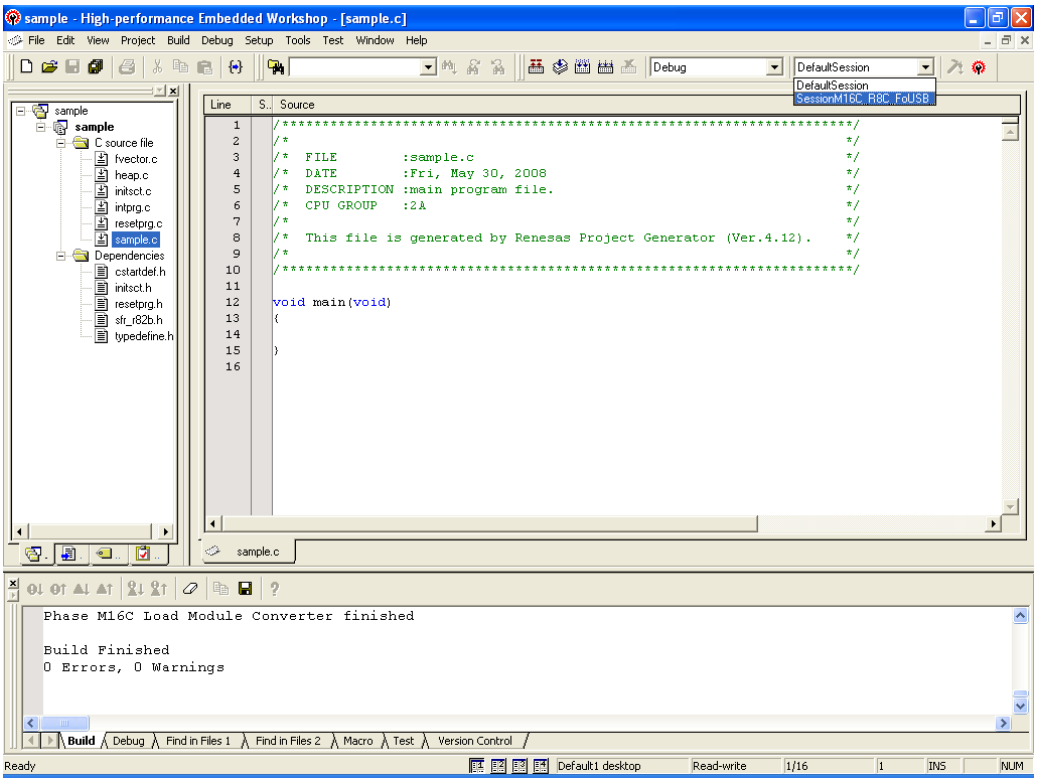
n) When the program is complete, under the “Build” menu, select “Build” or “Build All” to build the program.



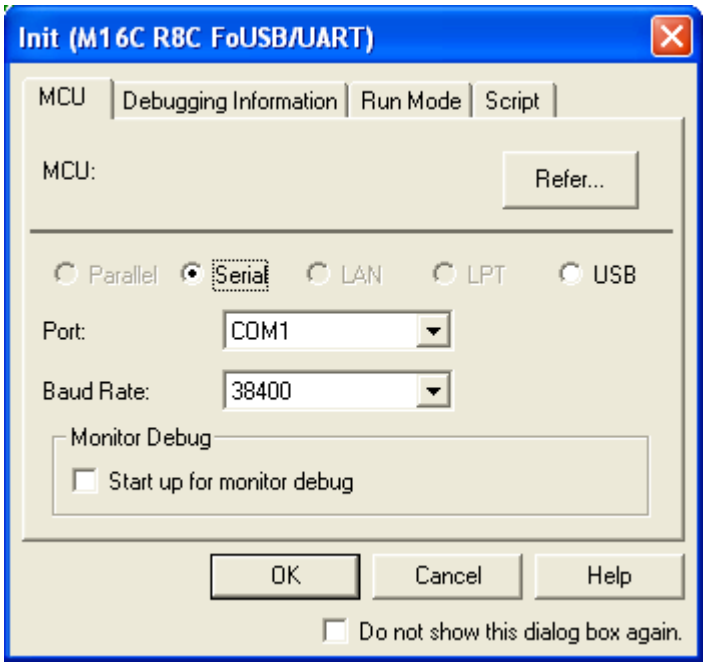
o) The result of the build is displayed.



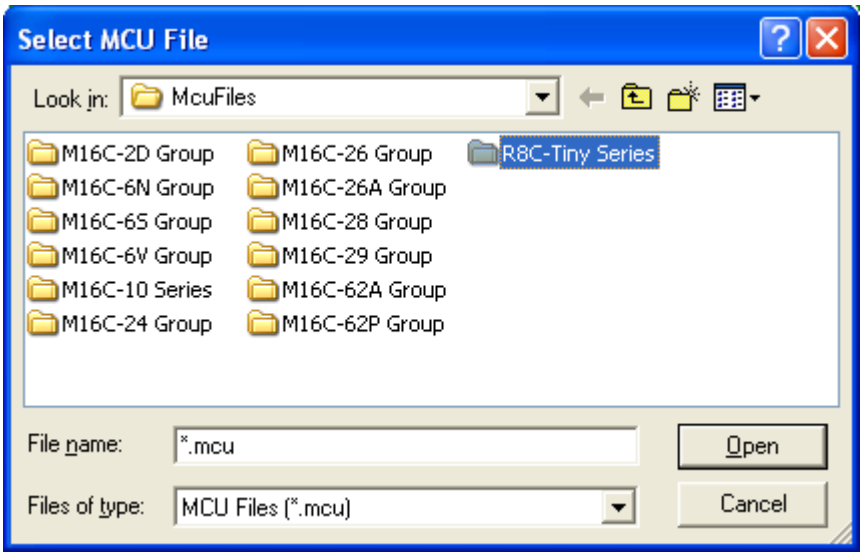
p) Connect with the target. A host computer can be connected with the target easily by switching to the session file in which the setting to use the R8C UART debugger is pre-registered.



q) The Init screen is displayed. Select the “Serial” radio button and press [Refer(ence)].

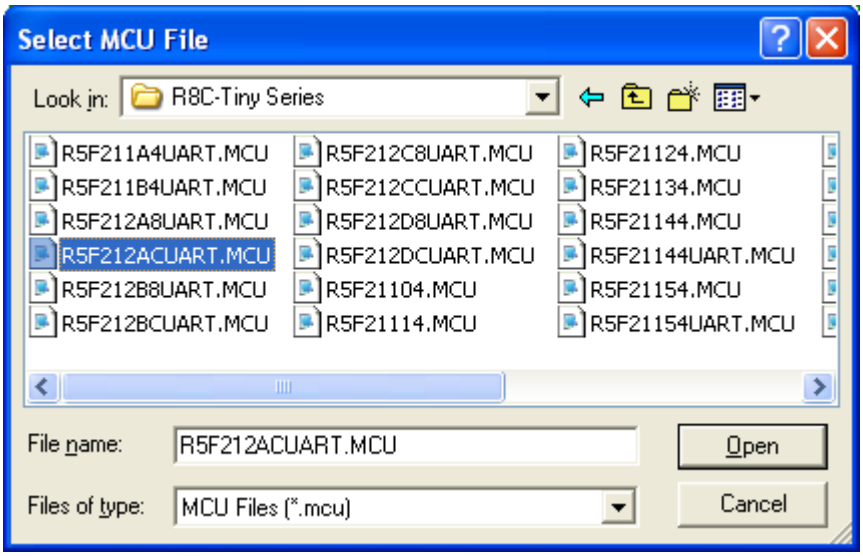


r) Select “R8C-Tiny Series”.

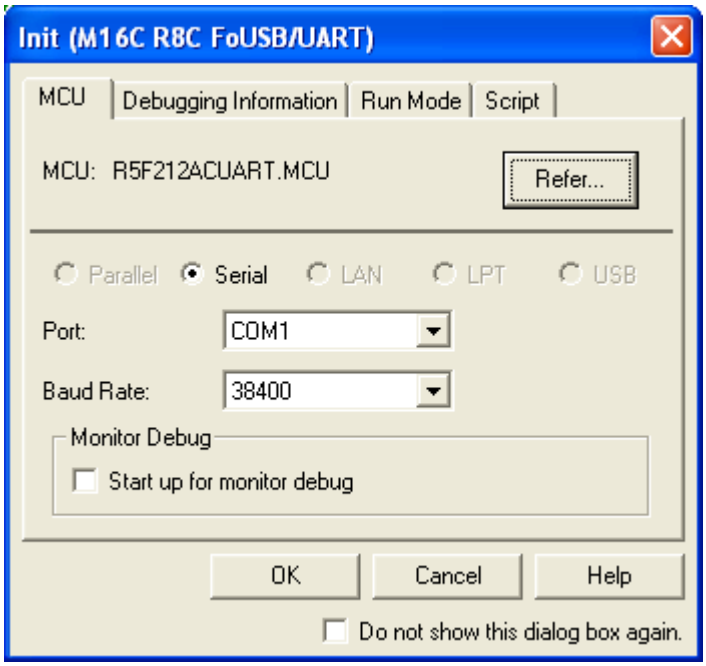


s) Select one of the following MCU files depending on the group and memory size.

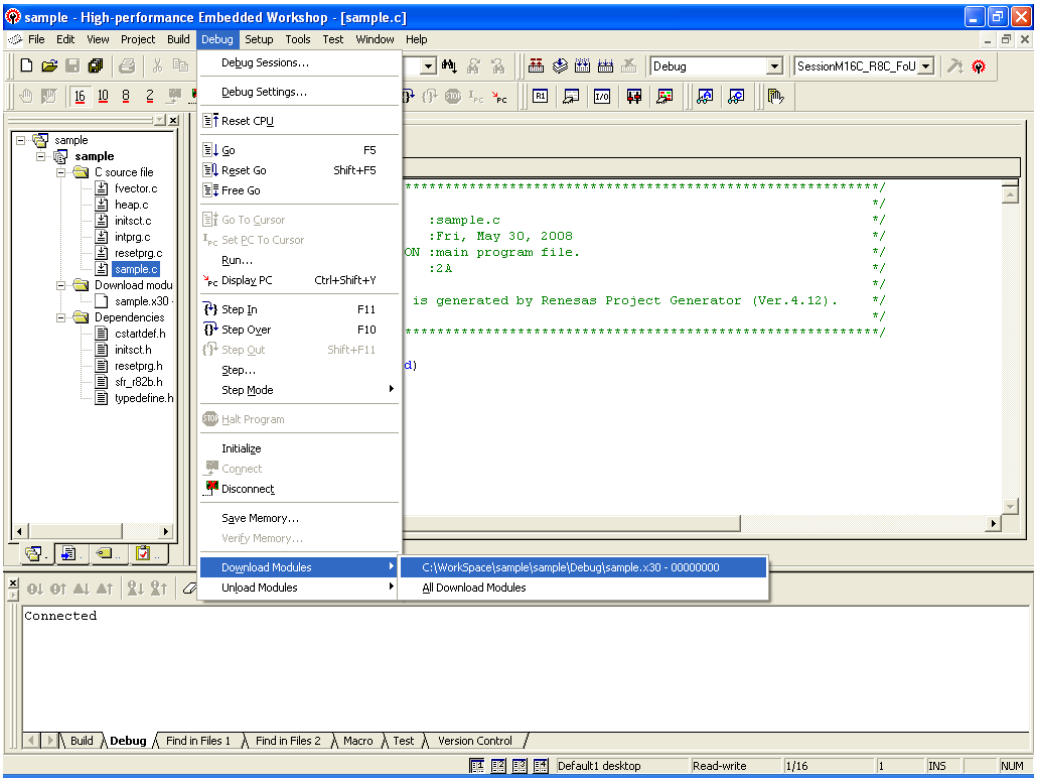
Group	Memory size 48KB, 64KB	Memory size 96KB, 128KB
R8C/2A	R5F212A8UART.MCU	R5F212ACUART.MCU
R8C/2B	R5F212B8UART.MCU	R5F212BCUART.MCU
R8C/2C	R5F212C8UART.MCU	R5F212CCUART.MCU
R8C/2D	R5F212D8UART.MCU	R5F212DCUART.MCU



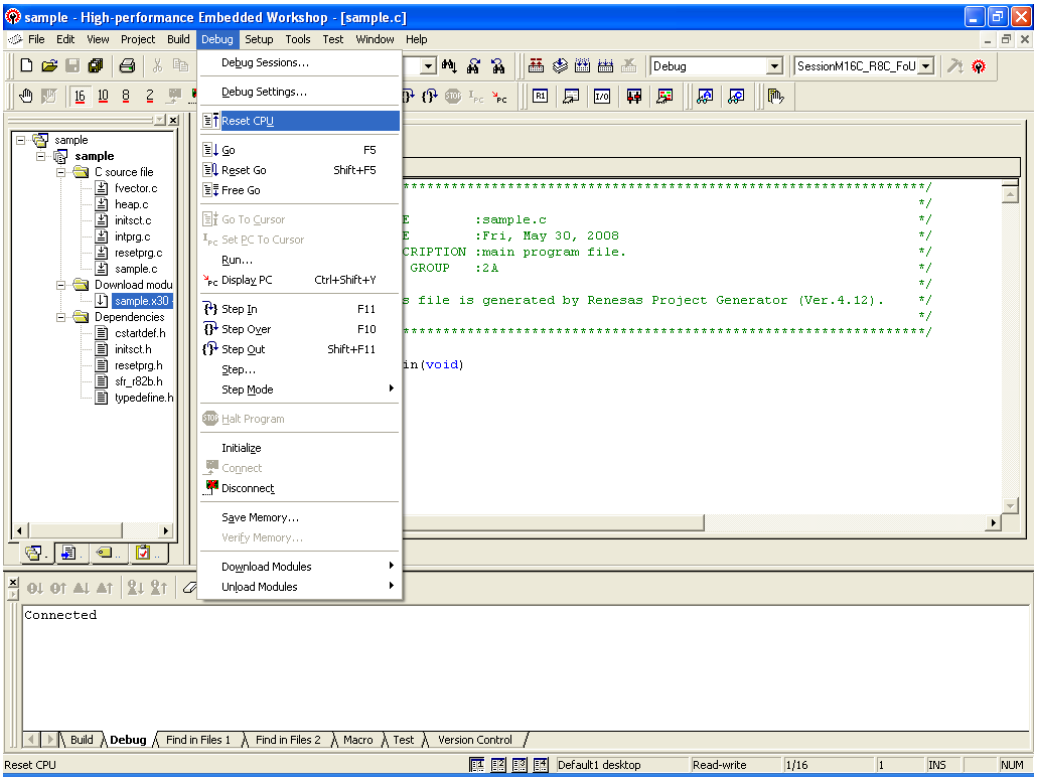
t) Select the appropriate ”Port” and ”Baud Rate” from the pull-down menus. Press [OK] and a monitor program is downloaded.



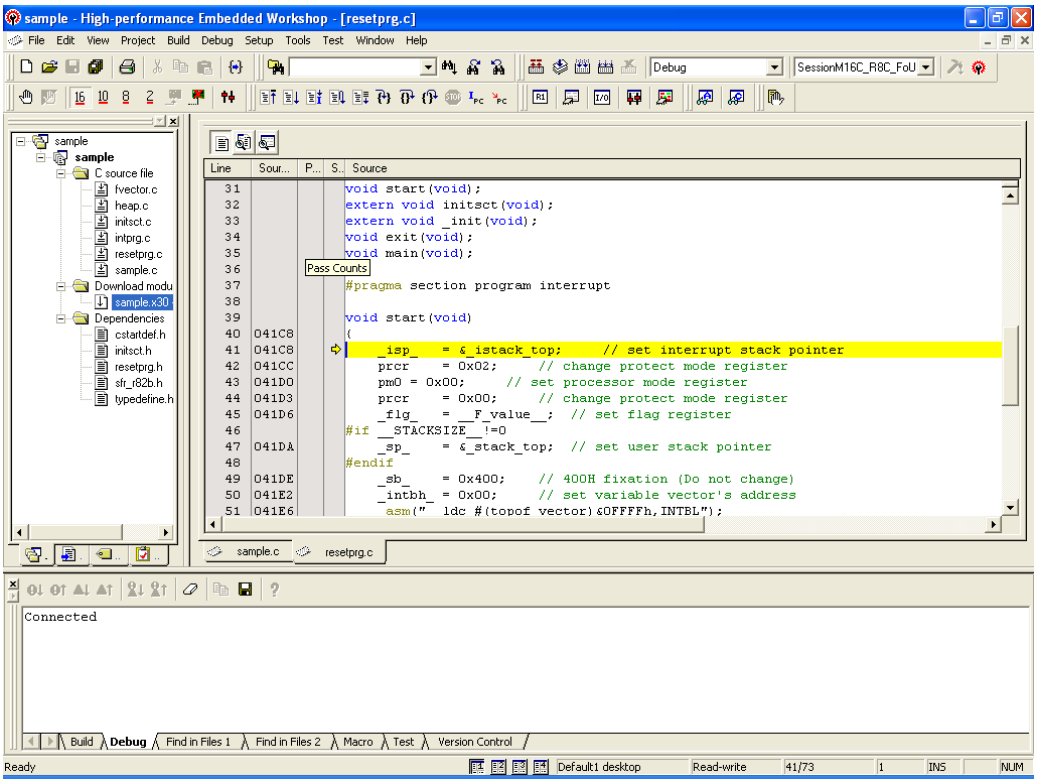
u) To download a user program, select “Download File (X30 file)” in the “Download Modules” submenu under the “Debug” menu.



v) To reset the user program, select “Reset CPU” under the “Debug” menu.



w) The cursor moves to the top of the user program and debugging starts.



3. Memory Map When Using the R8C UART Debugger

Figure 4 shows a memory map (64 KB).

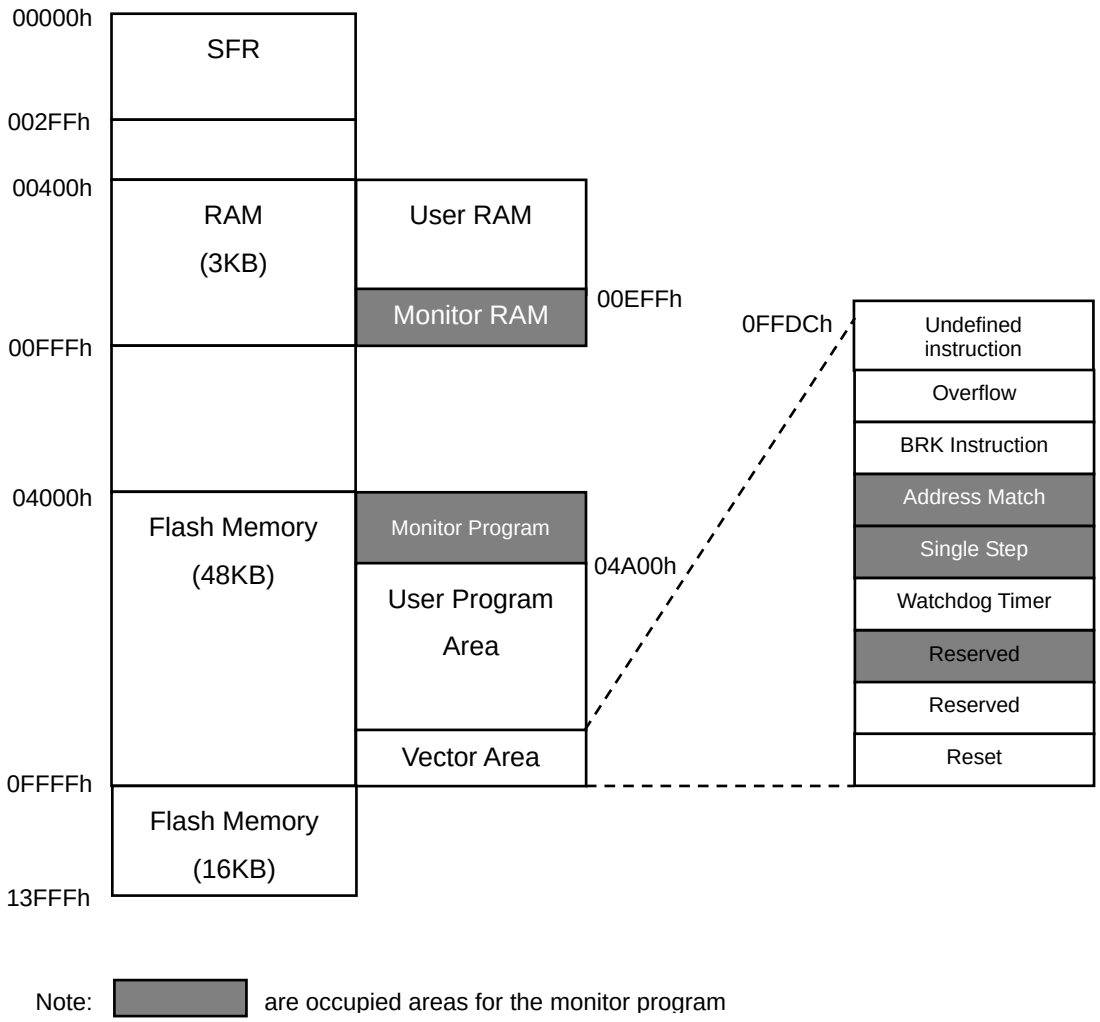


Figure 4 Memory Map (64 KB)

Figure 5 shows a memory map (128 KB).

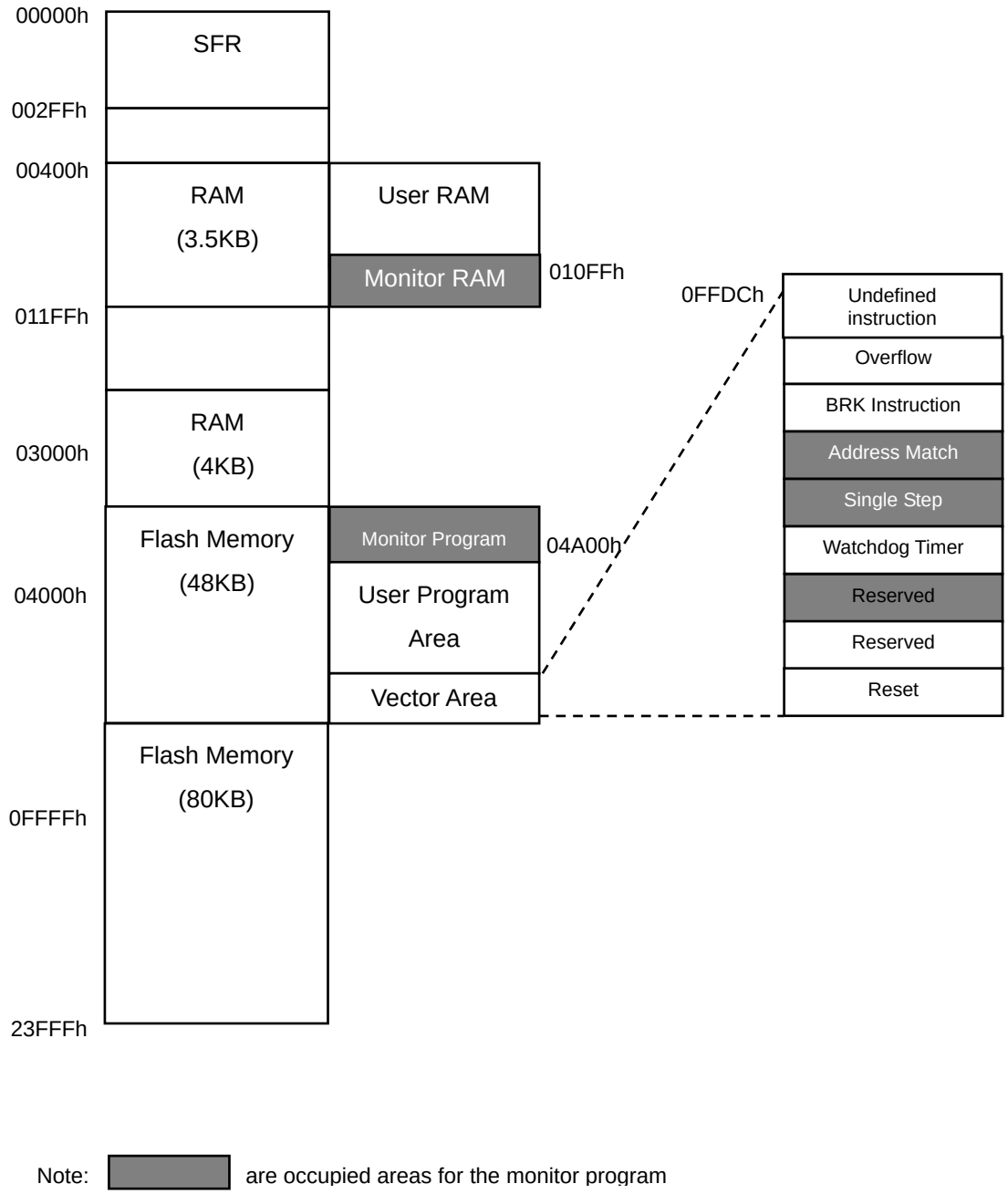


Figure 5 Memory Map (128 KB)

4. Monitor Program Occupied Area

Table 1 Monitor Program Occupied Area

ROM / RAM	Occupied Area for Monitor Program
48KB / 2.5KB	Flash memory 4000h to 49FFh Vector FFE8h to FFEBh, FFECh to FFEFh, FFF4h to FFF7h
64KB / 3KB	RAM EFFh to FFFh Flash memory 4000h to 49FFh Vector FFE8h to FFEBh, FFECh to FFEFh, FFF4h to FFF7h
96KB / 7KB	RAM 10FFh to 11FFh Flash memory 4000h to 49FFh Vector FFE8h to FFEBh, FFECh to FFEFh, FFF4h to FFF7h
128KB / 7.5KB	RAM 10FFh to 11FFh Flash memory 4000h to 49FFh Vector FFE8h to FFEBh, FFECh to FFEFh, FFF4h to FFF7h

5. Notes on Using the R8C UART Debugger

5.1. Changing communication speed and restarting the R8C UART debugger after the R8C UART debugger is done

The target MCU holds the baud rate value after the R8C UART debugger ends. Therefore, when changing the communication speed and restarting the R8C UART debugger, a communication error occurs. (The R8C UART debugger can be started when using the previous communication speed). When changing the communication speed, turn off the target power and turn on the power again.

5.2. User program ID code

Set the ID code of the user program to **all FFh** when using the R8C UART debugger.

Table 2 ID Code Storing Address

Address	ID No.	Vector Table
0FFDFh – 0FFDCh	ID1	Undefined instruction
0FFE3h – 0FFE0h	ID2	Overflow
0FFE7h – 0FFE4h		BRK instruction
0FFEBh – 0FFE8h	ID3	Address match
0FFEFh – 0FFECh	ID4	Single step
0FFF3h – 0FFF0h	ID5	Watchdog timer, oscillation stop detection, voltage monitor 2
0FFF7h – 0FFF4h	ID6	Reserved
0FFFBh – 0FFF8h	ID7	Reserved
0FFFFh – 0FFFCh	(See Note)	Reset

Note: Refer to the hardware manual for the value set to address 0FFFFh.

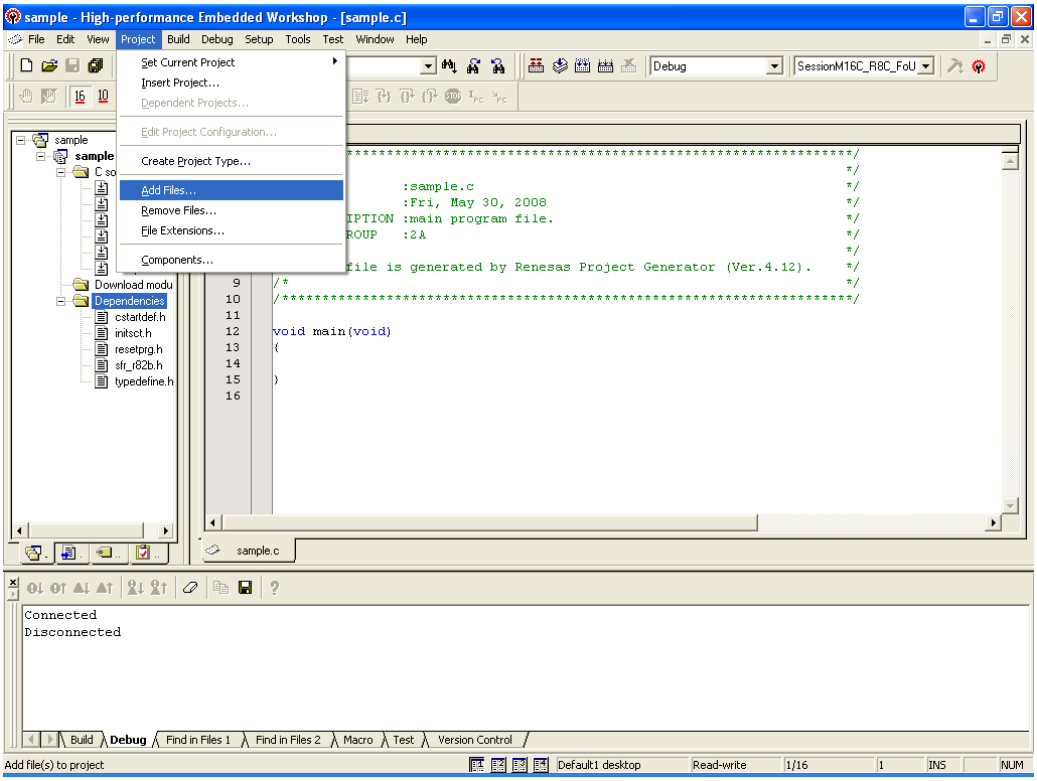
5.3. User program download area

As shown in Figures 4 and 5, a monitor program uses a part of RAM or flash memory when using the R8C UART debugger. The R8C UART debugger does not download a user program only in the area which overlaps with a monitor program. Note that the R8C UART debugger does not output an error at this time. When an error is output, set the following.

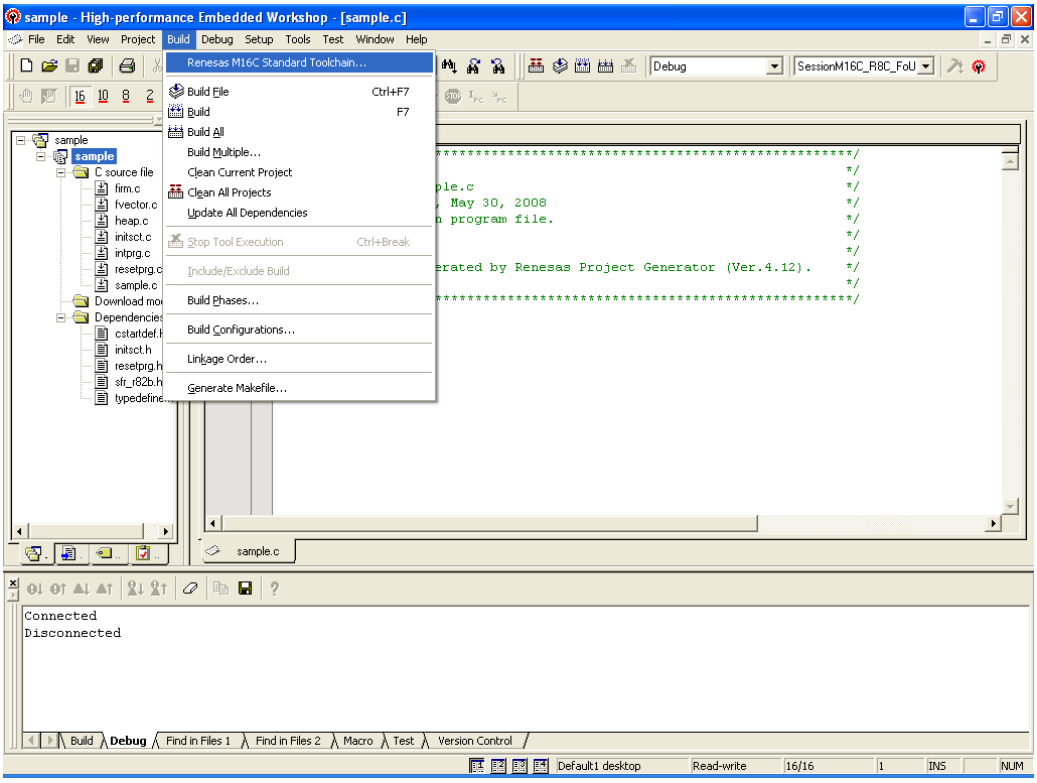
a) Make “firm.c” and enter the following:

```
#include "typedefine.h"
#ifdef __UART__

#pragma section bss FirmArea
_far _UBYTE _firmarea[0xA00];

#endif
```

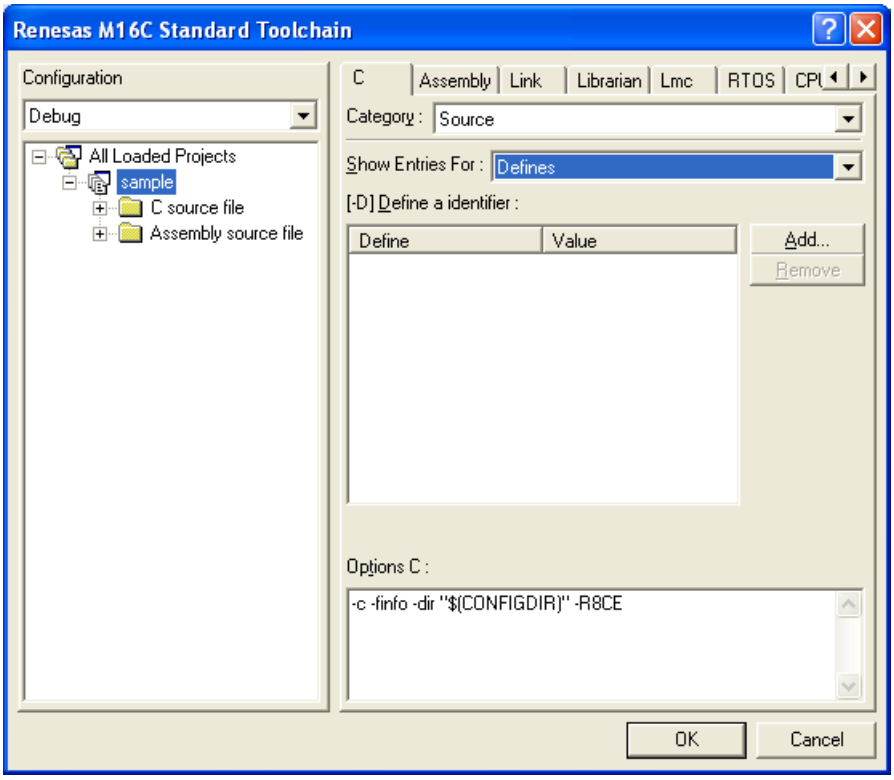
b) Add the file created above to the project. Under the “Project” menu, select “Add Files…”. When the file selection screen is displayed, select “firm.c”.



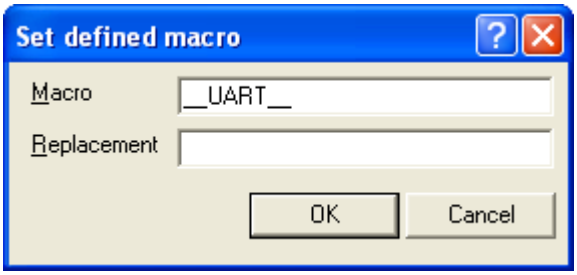
c) To add the compile option, select “Renesas M16C Standard Toolchain…” under the “Build” menu.



d) “Renesas M16C Standard Toolchain” is displayed. Select “Defines” from the “Show Entries For” pull-down menu, and then press [Add…].

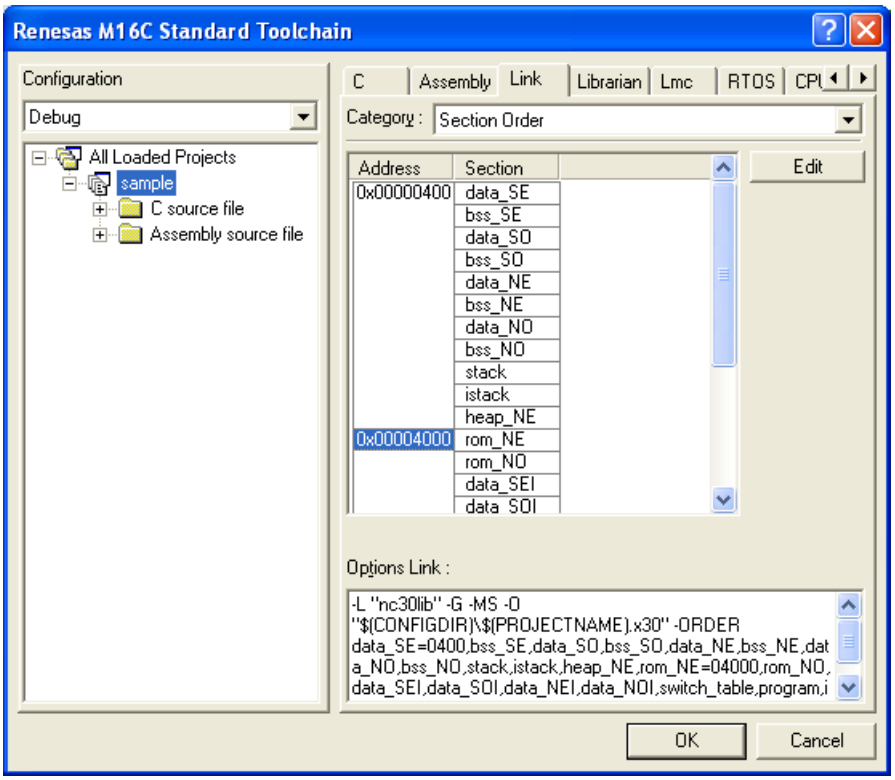


e) “Set defined macro” is displayed. Enter “_UART_” in the “Macro” field and press [OK[



f) To set the link option, in the “Renesas M16C Standard Toolchain” window, click on the “Link” tab and select “Section Order” from the “Category” pull-down menu. Add followings to the start address of the monitor program occupied area shown in Table 1.

- Memory size: 48KB
  FirmArea_NE
- Memory size: 64KB, 96KB, 128KB
  FirmArea_FE



When the user program overlaps with the monitor program occupied area due to the above settings, an error is output.

5.4. Frequency characteristics

The monitor program operates in the range of the main clock (Xin) frequency shown below. The monitor program may not run with frequencies not listed below, so use an oscillator that has this frequency range.

1 MHz (minimum) to 20 MHz (maximum)

Table 3 lists each frequency and communication available speed. However, note that operation may not be possible when dividing the main clock and using it with less than 1 MHz even in the range of the above frequency.

Table 3 Communication Available Speed of Each Frequency

Frequency	Communication Speed (bps)					
	1200	2400	4800	9600	19200	38400
20MHz	X	X	O	O	O	O
16MHz	X	X	O	O	O	O
14MHz	X	X	O	O	O	O
12MHz	X	X	O	O	O	O
10MHz	X	O	O	O	O	O
8MHz	X	O	O	O	O	O
6MHz	X	O	O	O	O	O
4MHz	O	O	O	O	O	X
2MHz	O	O	O	O	X	X
1MHz	O	O	O	X	X	X

O: Communication available

X: Communication not available

Note: Communication may not be possible depending on temperature and voltage. If communication is not possible, lower the communication speed.

5.5. Limitations of SFR operations

Table 4 lists the limitations of register operations. Changing registers that are disabled will cause the monitor program to malfunction.

Table 4 Limitations on SFR Operation

Register	Default Value	Limitation	Change
Processor Mode Register 0	Reset to 00h	Single-chip mode only	Partially enabled
Processor Mode Register 1	Reset to 00h	————————	Enabled
System Clock Control Register 0	Reset to 08h	Set the CM05 bit to 0.	Partially enabled
System Clock Control Register 1	Reset to 28h	Set bits CM13 and CM15 to 1. Set the CM14 bit to 0.	Partially enabled
High-Speed On-Chip Oscillator Control Register 0	Reset to 03h	Set the FRA00 bit to 1.	Partially enabled
High-Speed On-Chip Oscillator Control Register 1	————————	Do not change this register.	Disabled
High-Speed On-Chip Oscillator Control Register 2	Reset to 03h	————————	Enabled
Oscillation Stop Detection Register	Reset to 00000x00b	Set this register to 00h.	Partially enabled
Protect Register	————————	————————	Enabled
Flag Register	————————	Writing to the D flag is ignored. Do not set the D flag to 1.	Partially enabled
ISP (Interrupt Stack Pointer)	Reset to 057Fh	Set an area not used by the monitor program.	Partially enabled
UART 1 Transmit/Receive Mode Register	15h	Do not change this register.	Disabled
UART 1 Bit Rate Register	————————		
UART 1 Transmit/Receive Control Register 0	Reset to 000000xxb		
UART 1 Transmit/Receive Control Register 1	07h		
UART 1 Function Selection Register	00h		
Port Mode Register	10h		
UART 1 Transmit Buffer Register	————————	Do not write data to this register.	Disabled
UART 1 Receive Buffer Register	————————	Do not read this register.	Disabled

5.6. Limitations on stop mode or wait mode

When using stop mode or wait mode, start the R8C UART debugger in free-run mode, and close the RAM window, C watch window, and ASM window in advance. Also, do not operate the R8C UART debugger until the program stops at the break point by setting the break point after exiting stop mode or wait mode.

5.7. Watchdog timer

The watchdog timer is refreshed while the monitor program is running. When using the watchdog timer while running the user program, note that by referring to or changing memory content, the monitor program intervenes and the watchdog timer is refreshed.

5.8. Real-time operation of user program

- Sampling run mode (also known as sampling mode)

In sampling mode, execution status of the user program will be regularly monitored when executing Go and Come. Therefore, it is possible to detect when the user program is stopped by a break or other command. Select this mode when performing a normal debug.

- Free run mode

In free run mode, execution status of the user program will not be monitored when executing Go and Come. Although real-time operation of the user program is secure, it is not possible to detect if the user program is stopped by a break or other command. Therefore, even when the user program stops, the R8C UART debugger does not stop executing Go and Come. Press STOP to stop the R8C UART debugger.

Note: In free run mode, use the R8C UART debugger while the RAM window, C watch window, and ASM window are closed.

5.9. Executing anomalistic steps

- Software interrupt instruction

  The step execution cannot be performed continuously to the instruction internal process of the instructions (undefined instruction, overflow, BRK instruction, and INT instruction) which generate the software interrupts.

Example: INT instruction

```
        NOP
        NOP
        INT     #3          ↓  INT instruction is skipped over
        NOP                    when performing step execution
        JMP     MAIN
INT_3:
        NOP     ←  Address at which program
        NOP        execution should stop
        NOP
        REIT
```

- INT instruction

  To debug the program using the INT instruction, set the software break for the INT instruction process and use the Go command.

Example:

```
        NOP                          ↓
        INT     #3                   Executed by the Go
        NOP                          command
        JMP     MAIN                 ↓
INT_3:  ←
        NOP     Break
        NOP
        REIT
```

5.10. Limitations on peripheral functions

UART1 is used for communication between the monitor program and the host computer. Do not use UART1 in the user program. Do not connect the pins below to other pins as they are used for communication with the host computer.

- R8C/2A, 2B Groups

  TxD1 (37 pin), RxD1 (38 pin)

- R8C/2C, 2D Groups

  TxD1 (45 pin), RxD1 (46 pin)

5.11. Limitations on flag register

When operating the flag register on the user program, execute the **FSET instruction and FCLR instruction** not to change the debug flag (D flag).

5.12. Operation on peripheral I/O during a break

Although an interrupt cannot be accepted during a break, peripheral I/O continues operating. For example, when stopping a user program by a break during operating a timer, the timer continues counting, but the timer interrupt cannot be accepted.

REVISION HISTORY

Rev.	Date	Page	Description Summary
1.00	Jul. 15, 2008	–	First edition issued

M16C R8C FoUSB/UART Debugger User Manual
Notes on Connecting R8C/2A, R8C/2B, R8C/2C, R8C/2D

Publication Date	Rev.1.00 Jul. 15, 2008
Published by:	Sales Strategic Planning Div. Renesas Technology Corp.
Edited by:	Renesas Solutions Corp.

© 2008. Renesas Technology Corp. and Renesas Solutions Corp., All rights reserved. Printed in Japan.

M16C R8C FoUSB/UART Debugger
User Manual
Notes on Connecting R8C/2A, R8C/2B, R8C/2C, R8C/2D

Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan